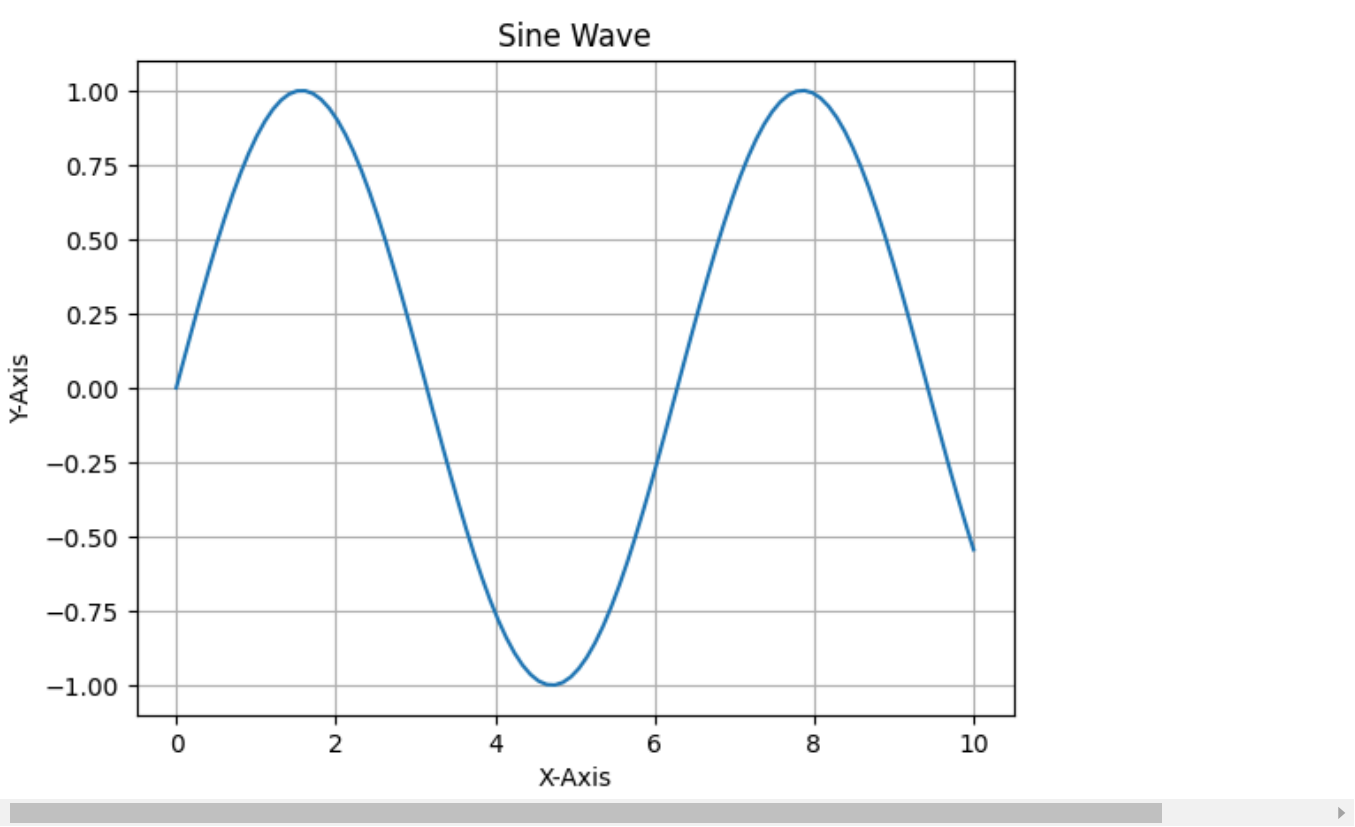
```
# Importing Matplotlib Library
import matplotlib.pyplot as plt
import numpy as np
```

1. Basic Line Plot

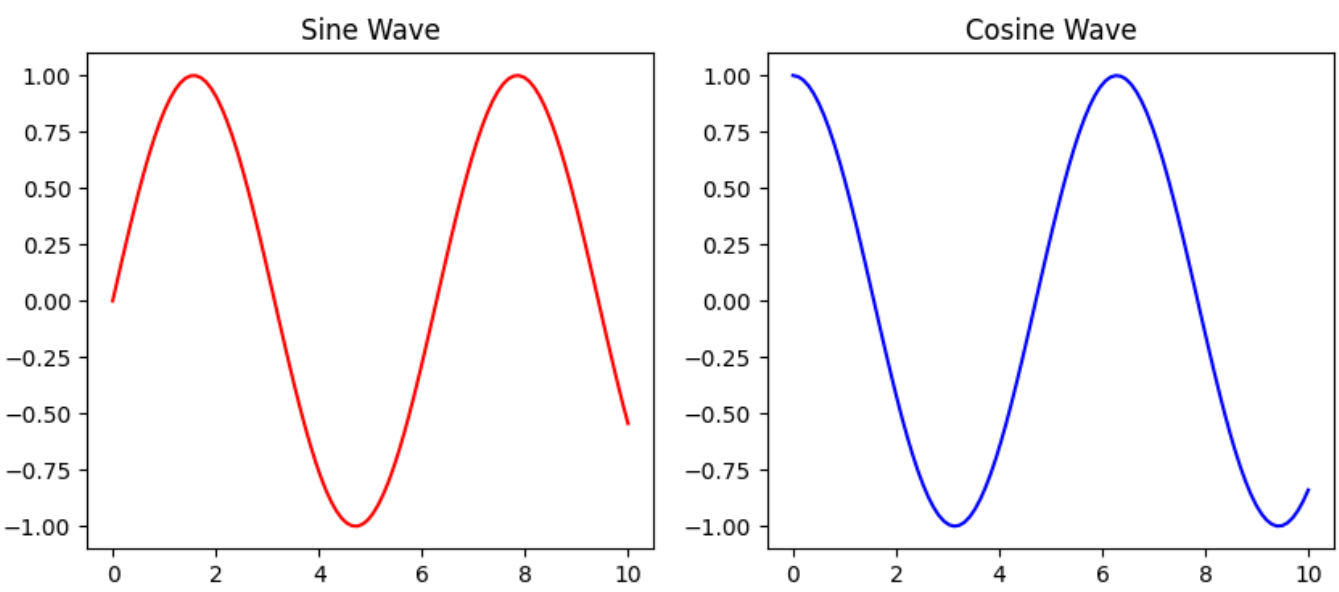
```
# Create a simple line plot
x = np.linspace(0, 10, 100)  # Generate 100 points between 0 and 10
y = np.sin(x)

plt.plot(x, y)  # Plot y = sin(x)
plt.title("Sine Wave")  # Add title
plt.xlabel("X-Axis")  # Add x-axis label
plt.ylabel("Y-Axis")  # Add y-axis label
plt.grid(True)  # Show grid
plt.show()  # Display the plot
```



2. Subplots: Multiple Plots in One Figure

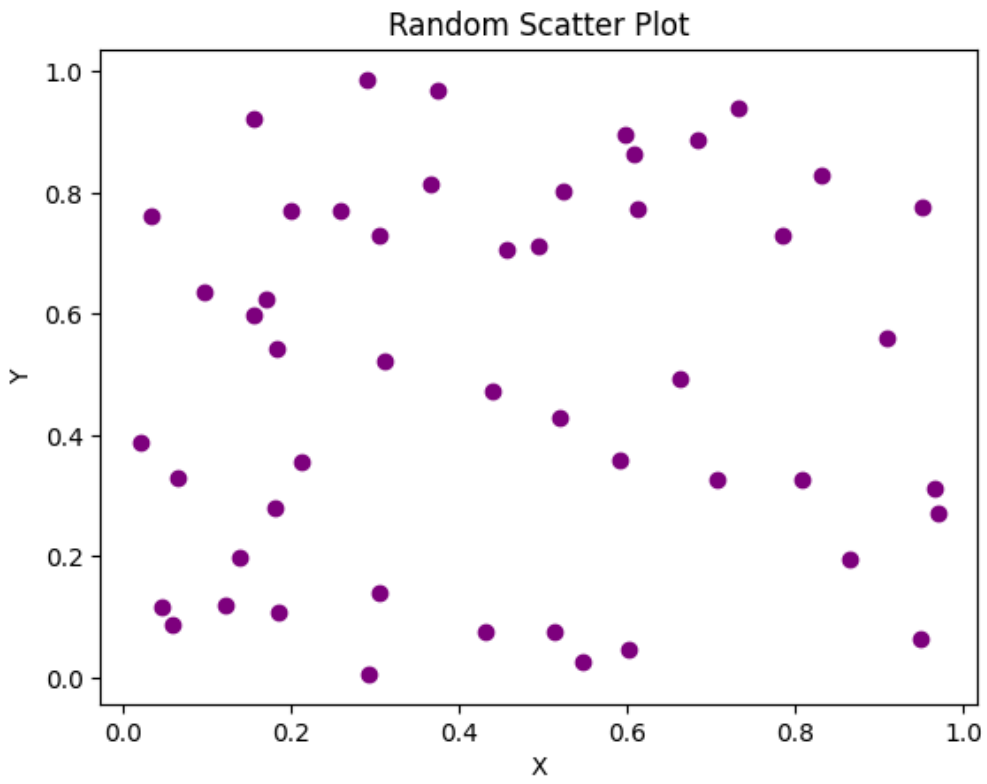
```
# Create multiple plots in one figure using subplots
x = np.linspace(0, 10, 100)
y1 = np.sin(x)
y2 = np.cos(x)
```

```
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(10, 4))  # Two side-by-side plots

ax1.plot(x, y1, 'r')  # Red sine wave
ax1.set_title("Sine Wave")

ax2.plot(x, y2, 'b')  # Blue cosine wave
ax2.set_title("Cosine Wave")

plt.show()
```



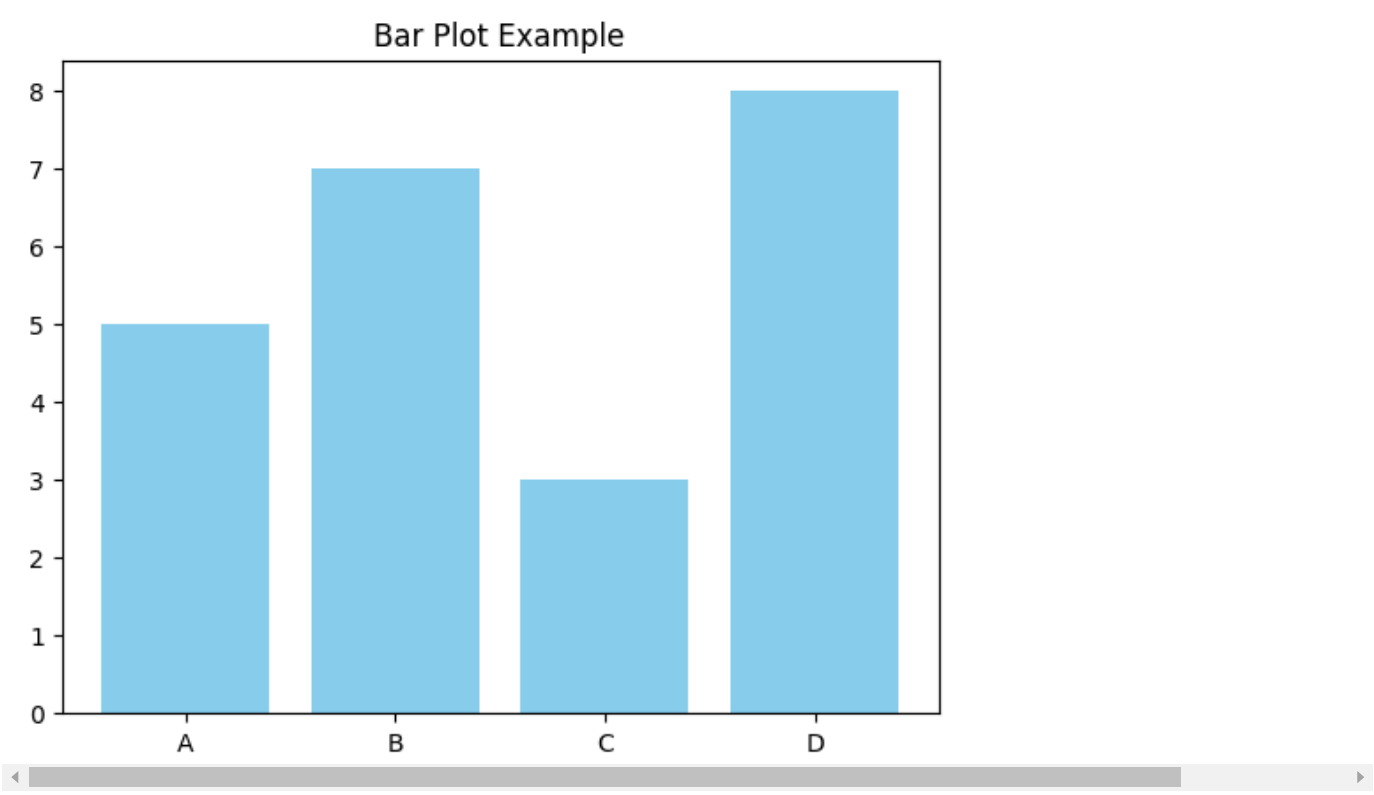
3. Scatter Plot

```
# Scatter plot with random data
np.random.seed(42)  # Seed for reproducibility
x = np.random.rand(50)
y = np.random.rand(50)

plt.scatter(x, y, c='purple', marker='o')
plt.title("Random Scatter Plot")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
```



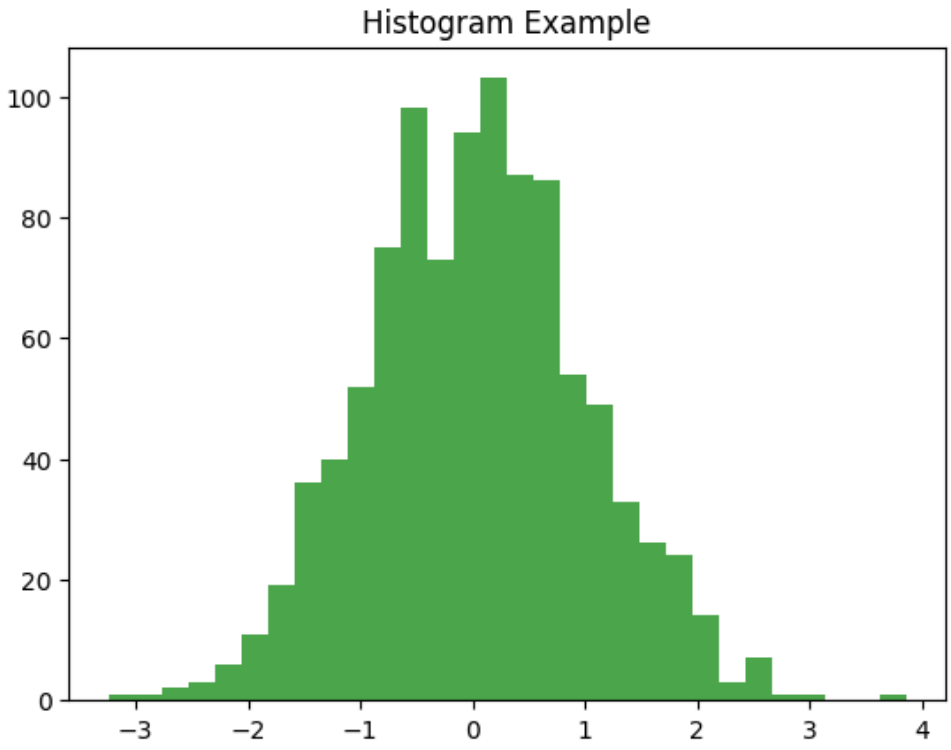
4. Bar Plot

```
# Bar plot to compare categories
categories = ['A', 'B', 'C', 'D']
values = [5, 7, 3, 8]

plt.bar(categories, values, color='skyblue')
plt.title("Bar Plot Example")
plt.show()
```



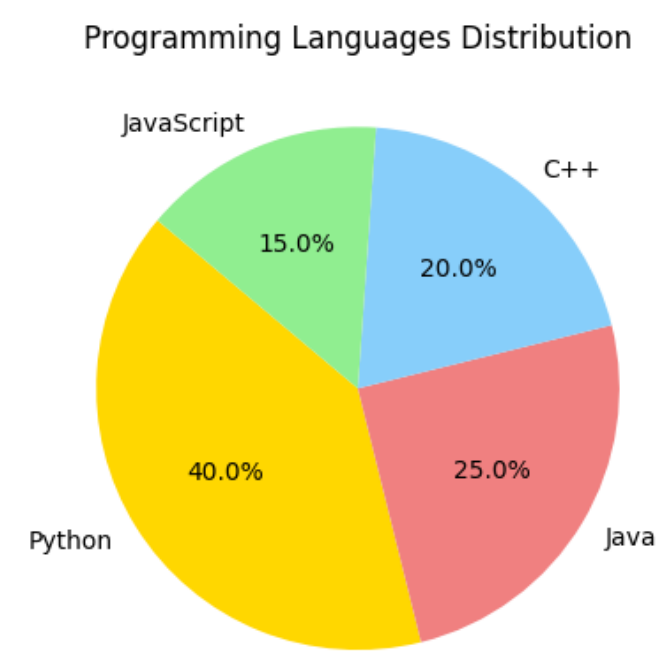
5. Histogram

```
# Histogram to show distribution
data = np.random.randn(1000)  # Generate 1000 random numbers from normal distribution

plt.hist(data, bins=30, color='green', alpha=0.7)
plt.title("Histogram Example")
plt.show()
```



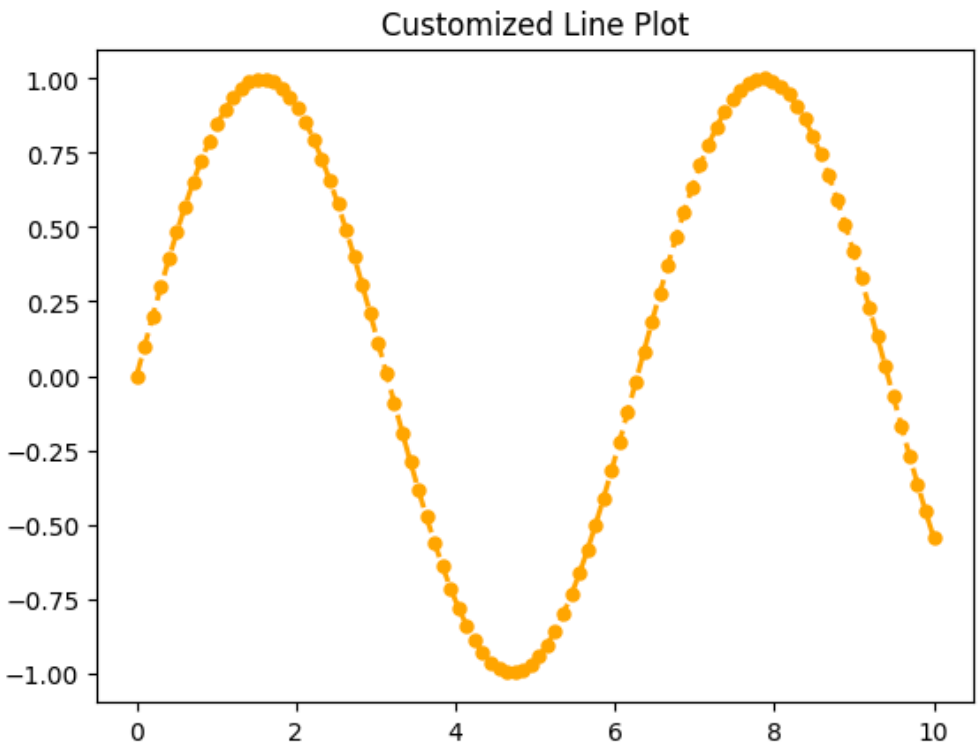
6. Pie Chart

```
# Pie chart to show proportions
labels = ['Python', 'Java', 'C++', 'JavaScript']
sizes = [40, 25, 20, 15]
colors = ['gold', 'lightcoral', 'lightskyblue', 'lightgreen']

plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%', startangle=140)
plt.title("Programming Languages Distribution")
plt.show()
```



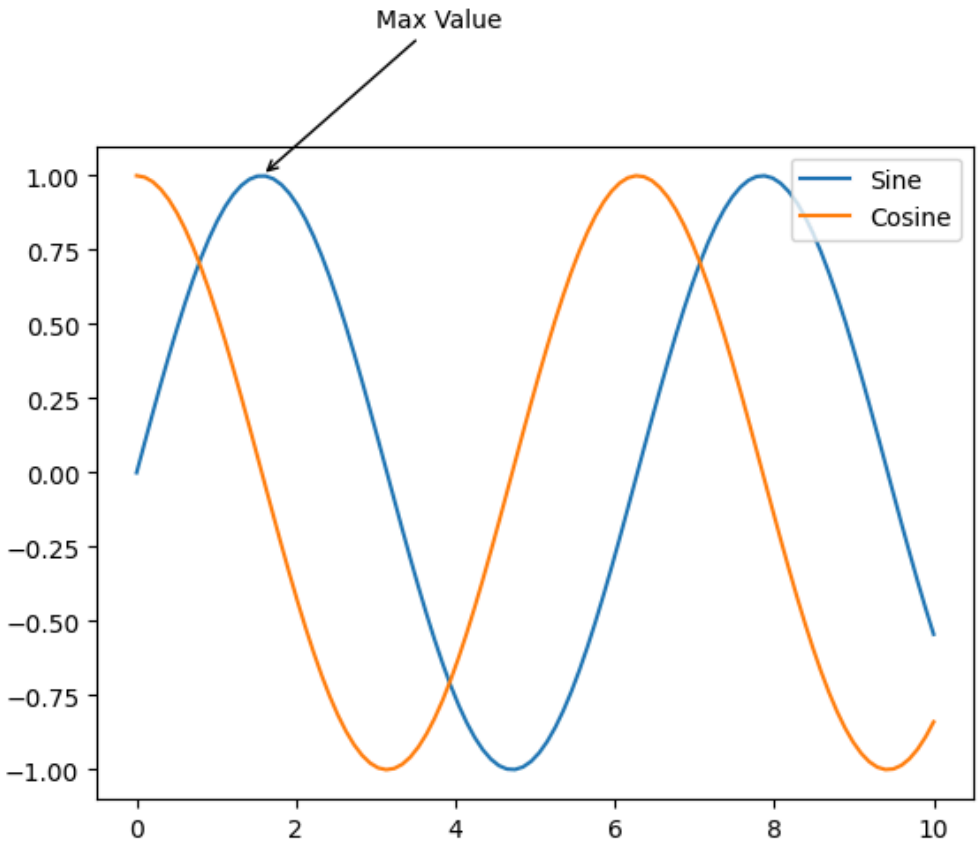
7. Customizing Plot Appearance

```
# Customize plot with colors, markers, and line styles
x = np.linspace(0, 10, 100)
y = np.sin(x)

plt.plot(x, y, color='orange', marker='o', linestyle='--', linewidth=2, markersize=5)
plt.title("Customized Line Plot")
plt.show()
```



8. Adding Legends and Annotations

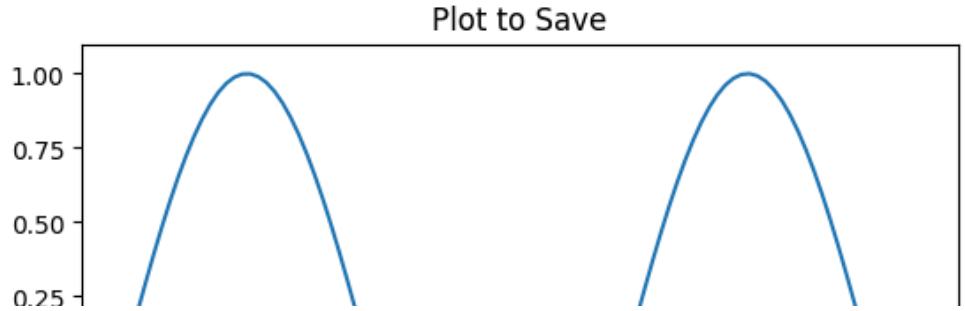
```
# Adding a legend and annotations
x = np.linspace(0, 10, 100)
y1 = np.sin(x)
y2 = np.cos(x)

plt.plot(x, y1, label='Sine')
plt.plot(x, y2, label='Cosine')
plt.legend(loc='upper right')  # Add legend

# Add annotation
plt.annotate('Max Value', xy=(1.57, 1), xytext=(3, 1.5),
             arrowprops=dict(facecolor='black', arrowstyle='->'))
plt.show()
```



9. Saving Plots

```
# Save plot as an image file
x = np.linspace(0, 10, 100)
y = np.sin(x)

plt.plot(x, y)
plt.title("Plot to Save")

plt.savefig('sine_wave.png')  # Save as PNG
print("Plot saved as 'sine_wave.png'")
plt.show()
```

```
Plot saved as 'sine_wave.png'
```



10. Advanced: Heatmap Using imshow

```
# Creating a heatmap using imshow
data = np.random.rand(5, 5)  # Generate random 5x5 matrix

plt.imshow(data, cmap='viridis', interpolation='nearest')
plt.colorbar()  # Add colorbar
plt.title("Heatmap Example")
plt.show()
```